DESCRIPTION OF COURSEWORK

Course Code	CST207
Course Name	Design and Analysis of Algorithms
Lecturer	Dr. Mohammed N. M. Ali Dr. Draib Najla'a Ateeq Mohammed
Academic Session	2025/09
Assessment Title	Project (Group) (20%)

A. Introduction/ Situation/ Background Information

This project is designed to test students' understanding and application of key concepts in Algorithms and Artificial Intelligence, including:

- Dataset Generation (Random, Nearly Sorted, Reversed, Few-Unique, Large Random)
- Sorting Algorithms (Bubble Sort, Insertion Sort, Merge Sort, Quick Sort)
- Performance Analysis (Execution time, comparisons, effect of dataset characteristics)
- AI-Based Algorithm Selection: using a Decision Tree or k-Nearest Neighbours (k-NN) model.
- Basic Feature Engineering: (dataset size, sortedness, unique elements)
- Integration of AI with Algorithmic Techniques

B. Course Learning Outcomes (CLO) covered

At the end of this assessment, students are able to:

CLO 3 Demonstrate problem-solving skills via concepts, techniques, and proficiencies in algorithm analysis which leads to potential applications in the self-development stage e.g., further studies.

C. University Policy on Academic Misconduct

1. Academic misconduct is a serious offense in Xiamen University Malaysia. It can be defined as any of the following:
   i. **Plagiarism** is submitting or presenting someone else's work, words, ideas, data or information as your own intentionally or unintentionally. This includes incorporating

published and unpublished material, whether in manuscript, printed or electronic form into your work without acknowledging the source (the person and the work).

ii. **Collusion** is two or more people collaborating on a piece of work (in part or whole) which is intended to be wholly individual and passed it off as own individual work.

iii. **Cheating** is an act of dishonesty or fraud in order to gain an unfair advantage in an assessment. This includes using or attempting to use, or assisting another to use materials that are prohibited or inappropriate, commissioning work from a third party, falsifying data, or breaching any examination rules.

2. All assessments submitted must be the student's own work, without any materials generated by AI tools, including direct copying and pasting of text or paraphrasing. Any form of academic misconduct, including using prohibited materials or inappropriate assistance, is a serious offense and will result in a zero mark for the entire assessment or part of it. If there is more than one guilty party, such as in case of collusion, all parties involved will receive the same penalty.

D. Instruction to Students

(1) Complete all tasks listed in Section F.

All required program components, AI modules, datasets, and documentation must be fully completed.

(2) Group Work Requirement: This assessment must be completed in **groups of up to 5 students**.

(3) Submission Guidelines: Only the **group representative** should upload the following files to Moodle:

- **Source Code**
  - **(i)** Submit the .c or .cpp file(s).
  - **(ii)** Ensure the code is clean, well-commented, and organized.
- **Project Report (PDF format)**: The report must include:
  - **(i) Cover pages** (project title, group name, student details)
  - **(ii) Contribution Table** outlining each group member's specific tasks
  - **(iii) Well-commented source code** (placed in an appendix)

(iv) The marking rubric (completed and included at the end of the report)

- **Oral Presentation Video**

(i) Submit an .mp4 file

(ii) Maximum duration: **7 minutes**

(iii) Must include:

- Overview of the system
- Demonstration of the program
- Each student clearly presents **their individual contribution**

(4) Submission Format:

- Upload all files as a **soft copy** to Moodle.
- Only **one submission per group** is required.

(5) Deadline: All files must be submitted by: **Friday, December 28, 2025, 6:00 PM**, *Late submissions will incur penalties.*

(6) Additional Notes:

- Only the group representative should upload the files.
- Use the following naming format for all submissions:
  YourGroupName_CST207_GroupProject_202509.pdf

E. Evaluation Breakdown

No.	Component Title	Percentage (%)
1.	Group Project	85
2.	Individual Work	15
	TOTAL	**100**

F. Task(s)

Coursework Title: AI-Driven Sorting Algorithm Optimizer

1. Project Scenario:

In modern software applications, sorting large datasets efficiently is critical. Different sorting algorithms perform differently depending on the dataset characteristics—size, order, and uniqueness of elements. Choosing the optimal sorting algorithm for a dataset can save time and computational resources.

In this project, you will design and implement a **C++/C program that can select the best sorting algorithm** for a given dataset using **Artificial Intelligence techniques**. Students will combine knowledge from **Design and Analysis of Algorithms** and **AI concepts** to solve this problem.

The project simulates a **real-world scenario**: imagine you are a software engineer designing a library that automatically picks the fastest sorting algorithm based on the dataset features.

2. Project Goal: The main goal of this project is to **create a program that can:**

1. Generate different types of datasets (Random, Nearly Sorted, Reversed, Few Unique, Large Random).
2. Implement multiple sorting algorithms (Bubble, Insertion, Merge, Quick Sort).
3. Measure the **execution performance** of sorting algorithms (comparisons and time).
4. Use an **AI module** to predict the **best sorting algorithm** for a given dataset.
5. Present performance results and AI predictions clearly.

Optional Extension: Students may implement their own AI module for predicting the best algorithm, such as a simple **k-NN**, **Decision Tree**, or other lightweight AI techniques.

3. Student Requirements: Each student or group must:

1. Implement at least **3 types of datasets**.
2. Implement **at least 3 sorting algorithms**.
3. Implement **an AI module** to select the best sorting algorithm automatically.
4. Measure and display:
    - Number of comparisons
    - Execution time (in milliseconds)
5. Display the **original array** for small datasets and show **performance statistics** for all sorting algorithms.
6. Predict the best algorithm using the AI module and compare it with actual performance.

AI Module Options:

- **Decision Tree:** Use dataset features (size, sortedness, unique elements) to predict the best sorting algorithm.
- **k-Nearest Neighbors (k-NN):** Predict based on training datasets with features and known best algorithms.
- **Custom AI Module:** Students can implement their own simple heuristic or learning-based module.

4. Suggested Dataset Types

1. **Random Dataset:** Elements generated randomly.
2. **Nearly Sorted Dataset:** Elements almost sorted with a few swaps.
3. **Reversed Dataset:** Elements sorted in descending order.
4. **Few Unique Dataset:** Dataset with repeated elements.
5. **Large Random Dataset:** Large dataset with randomly generated values.

5. Suggested Sorting Algorithms

1. **Bubble Sort** – Simple, works on small arrays.
2. **Insertion Sort** – Efficient for nearly sorted datasets.

3. **Merge Sort** – Stable, works for moderate to large datasets.
4. **Quick Sort** – Fast for large datasets.

Students may include additional algorithms if they want (optional).

6. AI Module Tasks

Decision Tree Approach

- Input: Dataset features (size, sortedness, unique elements).
- Traverse a manually or automatically built tree to predict the best algorithm.
- Output: Predicted best algorithm.

k-NN Approach

- Input: Dataset features.
- Compare features to training datasets.
- Use nearest neighbors (majority vote) to predict the best algorithm.
- Output: Predicted best algorithm.

Custom AI Approach (Optional)

- Students can implement their own AI/heuristic for prediction.
- Must be justified in the report.

7. Recommended Group Division (Up to 5 Students)

Student	Suggested Task
1	Implement dataset generation for all types
2	Implement sorting algorithms (Bubble, Insertion)
3	Implement advanced sorting algorithms (Merge, Quick) and measure performance
4	Implement an AI module (Decision Tree or k-NN) and integrate with sorting
5	Implement result visualization, console output, report generation, and testing

Note: For smaller groups, tasks can be combined. The table above is only a suggestion; you may divide the tasks differently.

8. Program Features to be Implemented

1. **User Input:**
    - Dataset type
    - Dataset size
    - Choice of AI module (Decision Tree / k-NN / Custom)
2. **Output:**
    - Original array (if small)
    - Number of comparisons and execution time for each algorithm
    - AI-predicted best algorithm
    - Actual best algorithm based on performance

9. Deliverables

1. **C++ / C Source Code**
    - Fully documented with comments
    - Correctly implements sorting algorithms and the AI module
2. **Report / Coursework Document**
    - **Introduction:** Explain project scenario and goals
    - **Algorithm Design:** Explain dataset generation, sorting algorithms, and AI approach
    - **Implementation:** Describe how the code works
    - **Results:** Tables showing performance and AI predictions
    - **Conclusion:** Discuss observations and AI accuracy
    - Optional: Screenshots of program output.

10. Notes for Students

- Focus on **correctness and clarity** of code.

- For AI, both Decision Tree and k-NN approaches are valid; students may explore additional simple AI techniques.
- Arrays larger than 1000 elements should **skip Bubble/Insertion sort** in actual testing to save execution time.
- Clearly **justify the AI predictions** and compare with actual performance in the report.

This setup ensures:

- Students understand **how dataset characteristics affect algorithm performance**.
- Students gain experience in **AI-assisted algorithm selection**.
- The project is **manageable in one month** and can be divided among 1–5 students.

APPENDIX 1

MARKING RUBRICS

Component Title	Group Work				Percentage (%)	
Criteria	Score and Descriptors					Weight (%)	Marks
	Excellent (5)	Good (4)	Average (3)	Need Improvement (2)	Poor (1)		
Dataset Generation **(CLO 3)**	All required datasets are implemented correctly	Minor missing datasets	Some datasets implemented	Many datasets missing	Dataset generation not implemented	10	
Sorting Algorithms **(CLO 3)**	All required algorithms implemented correctly and efficiently	Minor inefficiencies or small errors	Most algorithms implemented	Major issues or missing algorithms	Algorithms not implemented	20	
AI Module **(CLO 3)**	AI module correctly predicts best sorting algorithm	Minor issues in predictions or logic	AI module works partially	AI predictions mostly incorrect	AI module not implemented	20	
Performance Measurement **(CLO 3)**	All algorithms measured correctly with time & comparisons	Minor mistakes in measurements	Measurements mostly correct	Significant errors	No measurements	15	
Code Clarity & Documentation **(CLO 3)**	The code demonstrates excellent readability and clarity.	Minor readability issues	Some unclear code	Poorly documented	Code unreadable	10	
Report Quality **(CLO 3)**	Well-organized, complete report with clear analysis	Minor formatting or clarity issues	Adequate report with missing details	Poorly organized report	Report missing or unreadable	10	
						85	

Note to students: Please include the marking rubric when submitting your coursework.

Component Title	Individual Work				Percentage (%)		
Criteria	Score and Descriptors					Weight (%)	Marks
	Excellent (5)	Good (4)	Average (3)	Need Improvement (2)	Poor (1)		
Individual Contribution **(CLO 3)**	Clear, detailed description of individual work	Minor details missing	Contribution mentioned with little detail	Contribution vague or incomplete	Contribution not mentioned	10	
Presentation & Video **(CLO 3)**	Clear, well-paced, complete video	Minor pacing issues	Adequate video	Poor video clarity or incomplete	No video submitted	5	
						15	

Note to students: Please include the marking rubric when submitting your coursework.